REMARKS/ARGUMENTS

Applicants have received the Office action dated January 26, 2005, in which the Examiner: 1) rejected claims 1-4, 9-12, 17-19, 24-28 and 33-40 under 35 U.S.C. § 103(a) as being unpatentable over Carr et al. ("Compiler Optimizations for Improving Data Locality"); and 2) rejected claims 5-8, 13-16, 20-23 and 29-32 under 35 U.S.C. § 103(a) as being unpatentable over Carr et al. in view of McGehearty et al. (U.S. Pat. No. 5,797,013).

With this Response, Applicants have amended claims 1, 9, 17, 24-26, 33 and 35. Based on the amendments and arguments contained herein, Applicants respectfully request reconsideration and allowance of the pending claims.

I. § 103 REJECTIONS

Amended claim 1, in part, requires "analyzing an execution profile of the program after said distributing" and "based on the execution profile, determining whether to repeat said identifying a loop, said identifying each vector memory reference, said determining dependencies, and said distributing." Neither Carr nor McGehearty teaches or suggests these limitations.

Carr must be considered in its entirety, including disclosures that teach away from the claims (MPEP 2143.02). Specifically, Carr teaches a system that "only use[s] loop distribution to indirectly improve reuse by enabling loop permutation on a nest that is not permutable" (see pg. 256, section 4.4). The distribution algorithm implemented by Carr is only called "if memory order cannot be achieved on a nest and not all the inner nests can be fused" (see pg. 256, section 4.4). In other words, Carr teaches using loop distribution for a specified purpose that is different from the recitations of claim 1.

Carr does suggest "distribution could also be effective if there is no temporal locality between partitions and the accessed arrays are too numerous to fit in cache at once." Carr, however, specifically states that this issue in not addressed (see Footnote 3). Carr does not provide any additionally information regarding the use of distribution.

McGehearty teaches a cache collision avoidance mode that restructures loads and stores. However, none of the references cited by the Examiner, nor

combinations of the references, teach or suggest "analyzing an execution profile of the program after said distributing" and "based on the execution profile, determining whether to repeat said identifying a loop, said identifying each vector memory reference, said determining dependencies, and said distributing" as required in claim 1. For at least these reasons, Applicants submit that claim 1 and all claims that depend on claim 1 are allowable.

Amended claim 9, in part, requires that "said distributing the vector memory references into a plurality of detail loops is performed by a first computer for execution by a second computer." None of the references cited by the Examiner, nor combinations of the references, teach or suggest "distributing the vector memory references into a plurality of detail loops is performed by a first computer for execution by a second computer" as required in claim 9. For at least this reason, Applicants submit that claim 9 and all claims that depend on claim 9 are allowable.

Amended claim 17, in part, requires that "said identifying a loop, said identifying each vector memory reference, said determining dependencies between vector memory references and said distributing the vector memory references into a plurality of detail loops produce code that is substantially independent of a computer architecture" and "performing code optimizations that are dependent on a computer architecture after said distributing." None of the references cited by the Examiner, nor combinations of the references, teach or suggest "distributing the vector memory references into a plurality of detail loops produce[s] code that is substantially independent of a computer architecture" as required in claim 17. Further, none of the references cited by the Examiner, nor combinations of the references, appears to teach or suggest "performing code optimizations that are dependent on a computer architecture after said distributing" as required in claim 17. For at least these reasons, Applicants submit that claim 17 and all claims that depend on claim 17 are allowable.

Amended claim 24, in part, requires "temporary arrays [that] are configured to simultaneously fit in a single cache bank." The Examiner apparently equates Applicants claimed "temporary arrays" with registers taught in McGehearty.

However, McGehearty specifically teaches that registers and a cache are separate hardware components. Thus, McGehearty does not teach or suggest "temporary arrays [that] are configured to simultaneously fit in a single cache bank" as required in claim 24. None of the references cited by the Examiner, nor combinations of the references, teach or suggest this limitation. For at least this reason, Applicants submit that claim 24 and all claims that depend on claim 24 are allowable.

Amended claim 25, in part, requires "generating an expanded code of the program by distributing the vector memory references into a plurality of detail loops configured to allocate the vector memory references into temporary arrays that avoid cache synonyms, wherein the vector memory references that have circular dependencies therebetween are included in a common detail loop." Amended claim 25 also requires that "the expanded code is substantially independent of computer architectures." None of the references cited by the Examiner, nor combinations of the references, teach or suggest "generating an expanded code of the program by distributing the vector memory references into a plurality of detail loops...wherein the expanded code is substantially independent of computer architectures" as required in claim 25. For at least this reason, Applicants submit that claim 25 and all claims that depend on claim 25 are allowable.

Amended claim 26, in part, requires "means for determining an execution profile of the program after said distributing occurs" and "means for selectively repeating use of said means for identifying a loop, said means for identifying each vector memory reference, said means for determining dependencies, said means for distributing the vector memory references into a plurality of detail loops, and said means for determining an execution profile based on said execution profile." None of the references cited by the Examiner, nor combinations of the references, teach or suggest "means for selectively repeating use of said means for identifying a loop, said means for identifying each vector memory reference, said means for determining dependencies, said means for distributing the vector memory references into a plurality of detail loops, and said means for determining

an execution profile based on said execution profile" as required in claim 26. For at least this reason, Applicants submit that claim 26 and all claims that depend from claim 26 are allowable.

Amended claim 33, in part, requires "modifying the identified portion of the software to reduce the likelihood of cache thrashing by distributing cache synonyms into detail loops configured to allocate the cache synonyms into temporary storage areas, sized and located, to prevent cache thrashing." Also, claim 33 requires that "said modifying occurs before optimizations that are based on an architecture of the computer system." None of the references cited by the Examiner, nor combinations of the references, appear to teach or suggest "modifying...by distributing cache synonyms into detail loops" and "said modifying occurs before optimizations that are based on an architecture of the computer system" as required in claim 33. For at least these reasons, Applicants submit that claim 33 and all claims that depend on claim 33 are allowable.

Amended claim 35, in part, requires "temporary arrays [that] are configured to simultaneously fit in a single cache bank." Again, the Examiner apparently equates Applicants claimed "temporary arrays" with registers taught in McGehearty. However, McGehearty specifically teaches that registers and a cache are separate hardware components. Thus, McGehearty does not teach or suggest "temporary arrays [that] are configured to simultaneous fit in a single cache bank" as required in claim 35. None of the references cited by the Examiner, nor combinations of the references, teach or suggest this limitation. For at least this reason, Applicants submit that claim 35 and all claims that depend on claim 35 are allowable.

II. CONCLUSIONS

In the course of the foregoing discussions, Applicants may have at times referred to claim limitations in shorthand fashion, or may have focused on a particular claim element. This discussion should not be interpreted to mean that the other limitations can be ignored or dismissed. The claims must be viewed as a whole, and each limitation of the claims must be considered when determining the patentability of the claims. Moreover, it should be understood that there may

be other distinctions between the claims and the cited art which have yet to be raised, but which may be raised in the future.

Applicants respectfully request reconsideration and that a timely Notice of Allowance be issued in this case. It is believed that no extensions of time or fees are required, beyond those that may otherwise be provided for in documents accompanying this paper. However, in the event that additional extensions of time are necessary to allow consideration of this paper, such extensions are hereby petitioned under 37 C.F.R. § 1.136(a), and any fees required (including fees for net addition of claims) are hereby authorized to be charged to Hewlett-Packard Development Company's Deposit Account No. 08-2025.

Respectfully submitted,

Alan D. Christenson
PTO Reg. No. 54,036
CONLEY ROSE, P.C.
(713) 238-8000 (Phone)
(713) 238-8008 (Fax)
AGENT FOR APPLICANTS

HEWLETT-PACKARD COMPANY
Intellectual Property Administration
Legal Dept., M/S 35
P.O. Box 272400
Fort Collins, CO 80527-2400